**Description**

`recast` changes the storage type of the variables identified in `varlist` to `type`.

**Quick start**

Recast numeric variable `v1` to type `double` from any other numeric type
```
recast double v1
```
Recast string variable `v2` to `str30` from any length less than 30
```
recast str30 v2
```
As above, but for length longer than 30
```
recast str30 v2, force
```

**Syntax**

```
recast type varlist [, force]
```

where `type` is `byte`, `int`, `long`, `float`, `double`, `str1`, `str2`, ..., `str2045`, or `strL`.

**Option**

`force` makes `recast` unsafe by causing the variables to be given the new storage type even if that will cause a loss of precision, introduction of missing values, or, for string variables, the truncation of strings.

`force` should be used with caution. `force` is for those instances where you have a variable saved as a `double` but would now be satisfied to have the variable stored as a `float`, even though that would lead to a slight rounding of its values.

**Remarks and examples**

See [U] 12 Data for a description of storage types. Also see [D] compress and [D] destring for alternatives to `recast`. 
**Example 1**

`recast` refuses to change a variable’s type if that change is inappropriate for the values actually stored, so it is always safe to try:

```
use https://www.stata-press.com/data/r16/auto
(1978 Automobile Data)
```

```
. describe headroom
```

```
+-------------------------------------------------------------+
| variable | storage  | display | value | label                     |
|-----------|----------|---------|-------|---------------------------+
| headroom  | float    | %6.1f   |       | Headroom (in.)            |
|           |          |         |       |                           |
+-------------------------------------------------------------+
```

```
. recast int headroom
```

```
headroom: 37 values would be changed; not changed
```

Our attempt to change `headroom` from a `float` to an `int` was ignored—if the change had been made, 37 values would have changed. Here is an example where the type can be changed:

```
. describe mpg
```

```
+-------------------------------------------------------------+
| variable | storage  | display | value | label                     |
|-----------|----------|---------|-------|---------------------------+
| mpg       | int      | %8.0g   |       | Mileage (mpg)             |
|           |          |         |       |                           |
+-------------------------------------------------------------+
```

```
. recast byte mpg
```

```
. describe mpg
```

```
+-------------------------------------------------------------+
| variable | storage  | display | value | label                     |
|-----------|----------|---------|-------|---------------------------+
| mpg       | byte     | %8.0g   |       | Mileage (mpg)             |
|           |          |         |       |                           |
+-------------------------------------------------------------+
```

`recast` works with string variables as well as numeric variables, and it provides all the same protections:

```
. describe make
```

```
+-------------------------------------------------------------+
| variable | storage  | display | value | label                     |
|-----------|----------|---------|-------|---------------------------+
| make      | str18    | %-18s   |       | Make and Model            |
|           |          |         |       |                           |
+-------------------------------------------------------------+
```

```
. recast str16 make
```

```
make: 2 values would be changed; not changed
```

`recast` can be used both to promote and to demote variables:

```
. recast str20 make
```

```
. describe make
```

```
+-------------------------------------------------------------+
| variable | storage  | display | value | label                     |
|-----------|----------|---------|-------|---------------------------+
| make      | str20    | %-20s   |       | Make and Model            |
|           |          |         |       |                           |
+-------------------------------------------------------------+
```

---

**Also see**

[D] `compress` — Compress data in memory

[D] `destring` — Convert string variables to numeric variables and vice versa

[U] 12.2.2 Numeric storage types

[U] 12.4 Strings